REDES BAYESIANAS Y DECODIFICACIÓN

JOSÉ IGNACIO RONDA, SSR-ETSIT-UPM, JIR@GTI.SSR.UPM.ES

1. Introducción

Los códigos turbo [Berroux] se consideran la primera solución computacionalmente factible para alcanzar la capacidad de canal. Poco después de su publicación [Wiberg] se descubrió que el algoritmo de decodificación propuesto se podía ver como un caso particular de un algoritmo para calcular las probabilidades marginales de una variable aleatoria discreta multidimensional. Este algoritmo, conocido como *suma-producto* [Bishop], se basa en la representación de una función de probabilidad factorizada como un grafo bipartito en el que unos nodos representan variables y otros nodos representan factores.

El algoritmo suma-producto proporciona las probabilidades marginales exactas cuando este grafo no presenta bucles, pero en la práctica se puede utilizar también, en una versión iterativa, cuando existen bucles pero son largos, como en el caso de las funciones de probabilidad que representan las probabilidades a posteriori de los bits de los códigos turbo o los códigos LDPC.

En este documento proporcionamos el detalle de cómo el algoritmo suma producto da lugar a distintos algoritmos de decodificación propuestos anteriormente. Para ello vemos en primer lugar que su aplicación a la decodificación de un código convolucional corresponde al algoritmo BCJR, que es el bloque fundamental del algoritmo de decodificación de turbocódigos. A continuación comprobamos que la aplicación del algoritmo suma-producto a la decodificación de un turbocódigo con encadenamiento en paralelo corresponde al algoritmo de decodificación de turbocódigos

Finalmente consideramos la transmisión en un canal con interferencia entre símbolos y aplicamos el algoritmo suma-producto a la detección de una señal sin codificar para luego integrar el resultado en un esquema de turboecualización.

2. Códigos sin bucles

2.1. Un ejemplo sencillo. En esta sección obtenemos el algoritmo suma-producto para la obtención de las probabilidades a posteriori de los bits de un código [Wiberg, 3.1]. Todas las palabras del código tienen la misma probabilidad

$$P(X_1, \ldots, X_7) = \frac{\chi_C(X_1 \ldots, X_7)}{|C|}$$

donde C es el conjunto de palabras código, χ_C es una función que vale 1 uno si la palabra pertenece al código y cero en caso contrario y $|C|$ es el número de elementos de C, es decir, el número de palabras código.

Como el código está definido por las ecuaciones

$$X_1 \oplus X_2 \oplus X_4 = 0,$$
$$X_3 \oplus X_4 \oplus X_6 = 0,$$
$$X_4 \oplus X_5 \oplus X_7 = 0,$$

donde$\oplus$ representa suma módulo dos, tenemos

$$\chi_C(X_1 \ldots, X_7) \sim g_1(X_1, X_2, X_4) g_2(X_3, X_4, X_6) g_3(X_4, X_5, X_7),$$
$$g_1(X_1, X_2, X_4) = Z(X_1 \oplus X_2 \oplus X_4),$$
$$g_2(X_3, X_4, X_6) = Z(X_3 \oplus X_4 \oplus X_6),$$
$$g_3(X_4, X_5, X_7) = Z(X_4 \oplus X_5 \oplus X_7),$$

donde la función Z toma el valor 1 si su argumento vale cero y cero en caso contrario.

Una observación elemental de gran importancia para simplificar las operaciones es que una función de probabilidad

$$P(X_1, \ldots, X_N)$$

contiene la misma información que una función proporcional a ella

$$\tilde{P}(X_1, \ldots, X_N) = c\, P(X_1, \ldots, X_N),$$

puesto que, como

$$\sum_{X_1, \ldots, X_N} P(X_1, \ldots, X_N) = 1$$

podemos recuperar P a partir de $\tilde{P}$ mediante

$$P(X_1, \ldots, X_N) = \frac{1}{c} \tilde{P}(X_1, \ldots, X_N) = \frac{\tilde{P}(X_1, \ldots, X_N)}{\sum_{X_1, \ldots, X_N} \tilde{P}(X_1, \ldots, X_N)}$$

basta dividir entre la suma de todos sus valores para recuperar la función de probabilidad. Por ello trabajaremos con la función

$$\tilde{P}(X_1, \ldots, X_7) = g_1(X_1, X_2, X_4) g_2(X_3, X_4, X_6) g_3(X_4, X_5, X_7) \sim \tilde{P}(X_1, \ldots, X_7)$$

más adelante aplicaremos el mismo prinicipio a otras funciones de probabilidad para evitar cálculos inútiles y con frecuencia costosos. Los valores observados en el decodificador son

$$x_i = \alpha(-1)^{X_i} + n_i$$

donde α es un valor que suponemos conocido y las n_i son variables aleatorias gaussianas independientes de media nula y varianza σ^2. Las probabilidades a posteriori (es decir, teniendo en cuenta las observaciones) de las palabras código se calculan por la fórmula de Bayes:

$$\begin{aligned} P(X_1, \ldots, X_7 | x_1, \ldots, x_7) &= f(x_1, \ldots, x_7 | X_1, \ldots, X_7) P(X_1, \ldots, X_7) / f(x_1, \ldots, x_7) \\ &\sim f(x_1, \ldots, x_7 | X_1, \ldots, X_7) \tilde{P}(X_1, \ldots, X_7) \\ &= f(x_1 | X_1) \ldots f(x_7 | X_7) \tilde{P}(X_1, \ldots, X_7) \end{aligned}$$

donde

$$f(x_i|X_i) \sim \exp -\frac{(x_i - \alpha(-1)^{X_i})^2}{2\sigma_n^2}.$$

Esta función está representada en el grfo de la figura 2.1, en la que los círculos representan variables y los cuadrados factores. Para obtener las probabilidades a posteriori marginales de cada bit $p(x_i)$ tenemos que sumar respecto de las demás variables. Por ejemplo, para X_1 tenemos

$$p(x_1) \equiv P(X_1|x_1, \ldots, x_7) = \sum_{x_2 \cdots x_7} P(X_1, \ldots, X_7|x_1, \ldots, x_7)$$

y si en lugar de disponer de P tenemos únicamente una función proporcional a ella, lo que obtenemos será una función $\tilde{p}(x_1)$ proporcional a $p(x_1)$, que en este caso nos podremos permitir el lujo de calcular como

$$p(x_1) = \frac{\tilde{p}(x_1)}{\tilde{p}(0) + \tilde{p}(1)}.$$

Para aligerar la notación, en adelante dejaremos de distinguir entre funciones de probabilidad y funciones proporcionales a éstas, pero casi siempre estaremos en el segundo caso.

Vamos a realizar la suma correspondiente a la marginalización de x_1 intentando minimizar las operaciones a base de sacar factor común siempre que podamos. Obtenemos

$$\begin{aligned}
p(x_1) &= \sum_{X_2 \cdots X_7} P(X_1, \ldots, X_7|x_1, \ldots, x_7) \\
&= f(x_1|X_1) \sum_{X_2, X_4} \left[g_1(X_1, X_2, X_4) f(x_2|X_2) \left[f(x_4|X_4) C(x_4) D(x_4) \right] \right], \\
C(x_4) &= \sum_{X_3, X_6} f(x_3|X_3) f(x_6|X_6) g_2(x_3, x_4, x_6), \\
D(x_4) &= \sum_{X_5, X_7} f(x_5|X_5) f(x_7|X_7) g_3(x_4, x_5, x_7).
\end{aligned}$$

Para interpretar esta expresión consideremos la figura 2.1. Conviene visualizar el grafo como un árbol con raíz en el nodo correspondiente a la variable X_1. El algoritmo, conocido como *suma-producto* se puede describir como un paso sucesivo de mensajes entre nodos adyacentes. Los mensajes de un nodo variable x a un nodo factor F son funciones $\mu_{x \to F}(x)$ y los mensajes de un nodo factor F a un nodo variable x son funciones $\mu_{F \to x}(x)$.

En nuestro caso:

- Las hojas del árbol, que son todas nodos factor, y, como son hojas, son factores de una sola variable, envían su función al nodo al que están conectados.
- Cada nodo variable, cuando ha recibido los mensajes de todos los nodos factor a los que está conectados menos de uno, envía a éste un mensaje correspondiente al producto de las funciones que han recibido.

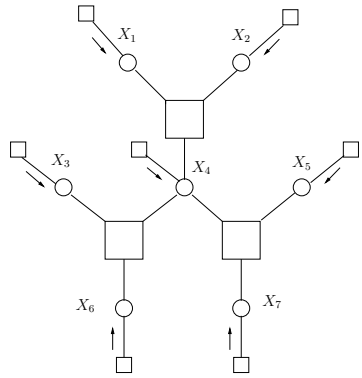

FIGURA 2.1. Esquema de paso de mensajes

- Cada nodo factor, cuando ha recibido los mensajes de todos los nodos variable a los que está conectado menos de uno envía a éste un mensaje correspondiente a la contracción de su función con las funciones recibidas.

Se puede demostrar [Wiberg] que este algoritmo está bien definido, termina en un número finito de pasos y proporciona las probabilidades marginales a posteriori de las variables como producto de los mensajes que le llegan.

2.2. Grafo con forma de cadena. Consideramos una función de probabilidad de la forma

$$P(s_1,\dots,s_N) = H_1(s_1)\left(\prod_{k=1}^{N-1} F_k(s_k,s_{k+1})\right) H_N(s_N).$$

Vamos a obtener directamente las probabilidades marginales de los estados., comprobando que el resultado equivale a la aplicación del algoritmo suma-producto. Tenemos, para $1 < k < N$,

$$P(s_1,\dots,s_N) = \underbrace{H_1(s_1)F_1(s_1,s_2)\cdots F_{k-1}(s_{k-1},s_k)}_{M_k(s_1,\dots,s_k)}\underbrace{F_k(s_k,s_{k+1})\cdots F_{N-1}(s_{N-1},s_N)H_N(s_N)}_{N_k(s_{k+1},\dots,s_N)}$$

$$\begin{aligned} P(s_k) &= \sum_{s_j\neq s_k} P(s_0,\dots,s_{N+1}) \\ &= \left(\sum_{(s_j),j<k} M_k(s_1,\dots,s_k)\right)\left(\sum_{(s_j),j>k} N_k(s_{k+1},\dots,s_N)\right). \end{aligned}$$

Notamos

$$\mu_{F_k\to s_k}(s_k) = \sum_{(s_j),j<k} M_k(s_1,\dots,s_k),$$

$$\mu_{F_{k+1}\to s_k}(s_k) = \sum_{(s_j),j>k} N_k(s_{k+1},\dots,s_N),$$

de forma que

$$P(s_k) = \mu_{F_k \to s_k}(s_k)\mu_{F_{k+1} \to s_k}(s_k) \tag{2.1}$$

Por otra parte,

$$\begin{aligned} M_k(s_1, \dots, s_k) &= M_{k-1}(s_1, \dots, s_{k-1})F_{k-1}(s_{k-1}, s_k) \\ \mu_{F_k \to s_k}(s_k) &= \sum_{(s_j), j<k} M_k(s_1, \dots, s_k) \\ &= \sum_{s_{k-1}} \left(\left(\sum_{(s_j), j<k-1} M_{k-1}(s_1, \dots, s_{k-1}) \right) F_{k-1}(s_{k-1}, s_k) \right) \\ &= \sum_{s_{k-1}} \mu_{F_{k-1} \to s_{k-1}}(s_{k-1})F_{k-1}(s_{k-1}, s_k) \end{aligned} \tag{2.2}$$

y, análogamente,

$$\mu_{F_{k+1} \to s_k}(s_k) = \sum_{s_{k+1}} \mu_{F_{k+2} \to s_{k+1}}(s_{k+1})F_{k+1}(s_k, s_{k+1}).$$

Efectivamente, el proceso es exactamente el correspondiente al algoritmo suma-producto, con la particularidad de que, al estar cada variable conectada únicamente a dos nodos, el mensaje que recibe de un nodo pasa directamente al otro sin multiplicarse por otros mensajes.

Las probabilidades de las transiciones $P(s_k, s_{k+1})$ se pueden obtener de la forma siguiente:

$$\begin{aligned} P(s_1, \dots, s_N) =& \underbrace{H_1(s_1)F_1(s_1, s_2) \cdots F_{k-1}(s_{k-1}, s_k)}_{M_k(s_1, \dots, s_k)} F_k(s_k, s_{k+1}) \\ & \underbrace{F_{k+1}(s_{k+1}, s_{k+2}) \cdots F_{N-1}(s_{N-1}, s_N)H_N(s_N)}_{N_{k+1}(s_{k+2}, \dots, s_N)} \end{aligned} \tag{2.3}$$

$$\begin{aligned} P(s_k, s_{k+1}) &= \sum_{s_j \neq s_k, s_{k+1}} P(s_0, \dots, s_{N+1}) \\ &= \left(\sum_{(s_j), j<k} M_k(s_1, \dots, s_k) \right) F_k(s_k, s_{k+1}) \left(\sum_{(s_j), j>k+1} N_{k+1}(s_{k+2}, \dots, s_N) \right) \\ &= \mu_{F_k \to s_k}(s_k)F_k(s_k, s_{k+1})\mu_{F_{k+2} \to s_{k+1}}(s_{k+1}). \end{aligned}$$

2.3. Grafo con forma de cadena con ramificaciones . Consideramos ahora una función de probabilidad de la forma

$$P = H_1(s_1)H_N(s_{N+1}) \prod_{k=1}^{N-1} p_k(x_k)G_k(x_k, s_k, s_{k+1}, y_k)q_k(y_k).$$

Se trata de una modificación de la estructura de cadena en la que hemos añadido a cada factor dos variables conectadas a él y a factores en los que figuna ellas solas. Hemos

añadido dos variables de este tipo a cada factor, pero podíamos haberlas agrupado en una de forma equivalente. Lo hacemos así para que el esquema guarde más similitud con los que aparecen en aplicaciones posteriores.

Para obtener las probabilidades marginales de los estados podemos primero marginalizar conjuntamente respecto de todas las s_k, es decir, sumar respecto de las x_k, y_k, y aplicar los resultados del apartado anterior. Los factores correspondientes resultan

$$F_k(s_k, s_{k+1}) = \sum_{x_k, y_k} p_k(x_k) G_k(x_k, s_k, s_{k+1}, y_k) q_k(y_k).$$

Para calcular las probabilidades marginales de los x_k, y_k partimos de la función de probabilidad

$$P(x_k, y_k, s_1, \ldots, s_N) = M_k(s_1, \ldots, s_k) p_k(x_k) G_k(x_k, s_k, s_{k+1}, y_k) q_k(y_k) N_{k+1}(s_{k+2}, \ldots, s_N)$$

donde hemos sumado respecto de las $x_j, y_j, j \neq k$ y utilizado la notación del apartado anterior. Sumando respecto de todas las s_j obtenemos

(2.4)

$$P(x_k, y_k) = p_k(x_k) q_k(y_k) \sum_{s_k, s_{k+1}} \left(\mu_{F_k \to s_k}(s_k) G_k(x_k, s_k, s_{k+1}, y_k) \mu_{F_{k+2} \to s_{k+1}}(s_{k+1}) \right).$$

3. Códigos convolucionales

3.1. Código sistemático recursivo. En esta sección comprobamos que el algoritmo BCJR para la decodificación de códigos convolucionales, tal como se detalla en [Madhow], corresponde a la aplicación del algoritmo suma-producto tal como se ha explicado en la sección anterior, que corresponde a la descripción de [Wiberg] y de [Bishop].

Consideramos la decodificación suave del código convolucional sistemático recursivo de tasa 1/2 dado por

$$Y_k \oplus Y_{k-1} \oplus Y_{k-2} = X_k \oplus X_{k-2}.$$

Las palabras código corresponden al diagrama de la figura 3.1, que representa un grafo de factorización en el que los círculos representan variables y los cuadrados factores. Cada factor vale uno si la suma módulo dos de las variables asociadas es cero. El producto de los factores es proporcional a la función de probabilidad de las palabras código. Como este grafo presenta bucles, no se le puede aplicar el algoritmo suma-producto.

Para obtener un grafo del código sin bucles introducimos las variables de estado s_k^0, s_k^1, correspondientes al contenido del registro de desplazamiento de la implementación usual del codificador. Las ecuaciones del codificador como máquina de estados son

$$\begin{aligned} Y_k &= X_k \oplus S_k^0 \\ S_{k+1}^0 &= X_k \oplus S_k^0 \oplus S_k^1 \\ S_{k+1}^1 &= S_k^0 \end{aligned}$$

El resultado es el esquema de la figura 3.2. Obsérvese que si consideramos los factores por separado también tenemos bucles, porque las variables x_k y s_k^0 aparecen en las dos primeras ecuaciones de arriba. Para solventar este problema basta agrupar estos dos

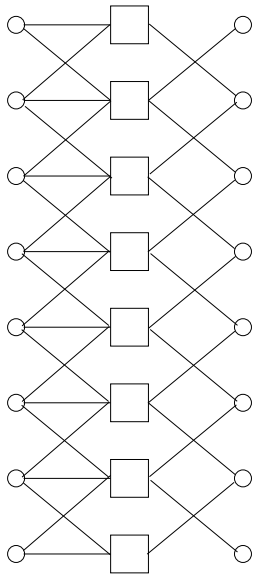

FIGURA 3.1. Diagrama de código convolucional. No se puede utilizar este modelo porque presenta bucles.

factores en uno, pero para simplificar la estructura general agrupamos los tres factores en uno, que queda

$$h(S_k^0, S_k^1, X_k, Y_k, S_{k+1}^0, S_{k+1}^1) = Z(Y_k \oplus X_k \oplus S_k^0) Z(S_{k+1}^0 \oplus X_k \oplus S_k^0 \oplus S_k^1) Z(S_{k+1}^1 \oplus S_k^0).$$

Además de esta forma obtenemos una estructura válida para cualquier modulación con memoria.

Vamos a aplicar a este código el algoritmo suma-producto. Para ello consideramos la figura 3.3. Para que el resultado sea aplicable al turbocódigo paralelo vamos a considerar también que los bits de entrada x_k tienen ciertas probabilidades a priori p_k. Notando $\mathbf{S}_k = (S_k^0, S_k^1)$, $\mathbf{S} = (\mathbf{S}_k)$, $\mathbf{X} = (X_k)$, $\mathbf{Y} = (Y_k)$, la función de probabilidad conjunta a posteriori es

$$P(\mathbf{S}, \mathbf{X}, \mathbf{Y}) = \prod_{k=1}^{N} f(y_k|Y_k) f(x_k|X_k) h_k(\mathbf{S}_k, \mathbf{S}_{k+1}, X_k, Y_k) p_k(X_k).$$

En los bloques del diagrama en los que faltan variables por efecto de la inicialización y la terminación en el estado cero basta sustituir estas variables por ceros.

Como en este codificador cada transición entre estados $(\mathbf{S}_k, \mathbf{S}_{k+1})$ corresponde a un único par de bits X_k, Y_k, podemos considerar los estados como únicas variables del modelo y trabajar con la función de probabilidad

$$P(\mathbf{S}) = \prod_{k=1}^{N} \gamma_k(\mathbf{S}_k, \mathbf{S}_{k+1})$$

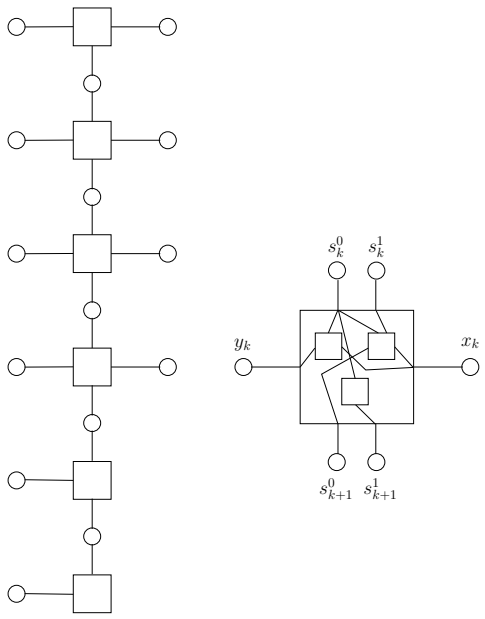

FIGURA 3.2. Diagrama de código convolucional con variables de estado.

donde

$$\begin{aligned}\gamma_k(\mathbf{S}_k, \mathbf{S}_{k+1}) =& f\left(y_k | Y(\mathbf{S}_k, \mathbf{S}_{k+1})\right) f\left(x_k | X(\mathbf{S}_k, \mathbf{S}_{k+1})\right) \\ & h_k\left(\mathbf{S}_k, \mathbf{S}_{k+1}, X(\mathbf{S}_k, \mathbf{S}_{k+1}), Y(\mathbf{S}_k, \mathbf{S}_{k+1})\right) p_k(X_k).\end{aligned}$$

Aplicamos el algoritmo suma-producto a este modelo, que corresponde a los bloques ampliados representados en línea de puntos. De esta forma obtendremos el algoritmo BCJR.

Notamos por $\alpha_k(\mathbf{S})$ los mensajes que envían los factores hacia abajo y por $\beta_k(\mathbf{S})$ los mensajes que envían hacia arriba. El mensaje del módulo superior al inferior es

$$\alpha_1(\mathbf{S}_1) = \gamma_1(\mathbf{0}, \mathbf{S}_1).$$

Este mensaje lo transfiere a la salida tal cual el nodo-variable $\mathbf{S}_1$. Un nodo genérico recibe el mensaje $\alpha_k(\mathbf{S}_k)$ y genera el mensaje

$$\alpha_{k+1}(\mathbf{S}_{k+1}) = \sum_{\mathbf{S}_k} \gamma_k(\mathbf{S}_k, \mathbf{S}_{k+1}) \alpha_k(\mathbf{S}_k)$$

que corresponde a (2.2). Análogamente, el nodo-factor N genera el mensaje

$$\beta_N(\mathbf{S}_N) = \gamma_N(\mathbf{S}_N, \mathbf{0})$$

y un nodo-factor genérico k que reciba del nodo-variable el mensaje $\beta_{k+1}(\mathbf{S}_{k+1})$ generará hacia el nodo-variable $\mathbf{S}_k$ el mensaje

$$\beta_k(\mathbf{S}_k) = \sum_{\mathbf{S}_k} \gamma_k(\mathbf{S}_k, \mathbf{S}_{k+1}) \beta_{k+1}(\mathbf{S}_{k+1}).$$

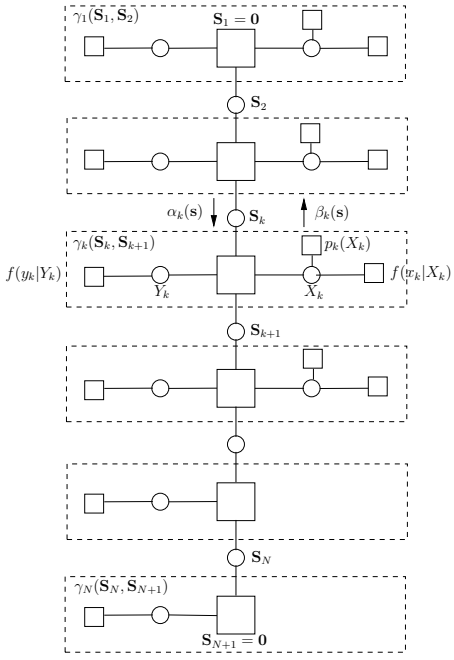

FIGURA 3.3. Aplicación del algoritmo suma-producto al código convolucional.

Este cálculo proporciona las probabilidades marginales a posteriori de los estados como

$$P_{\mathbf{S}_k}(\mathbf{S}_k) = \alpha_k(\mathbf{S}_k)\beta_k(\mathbf{S}_k)$$

como en (2.1). Para calcular las de los bits de información,X_k, bajamos al nivel de mayor detalle. El nodo-factor h_k recibe por su izquierda el mensaje

$$f(y_k|Y_k)$$

y genera hacia X_k el mensaje

$$\begin{aligned} m_k(X_k) = & \sum_{X(\mathbf{S}_k,\mathbf{S}_{k+1})=X_k,Y_k} h_k(\mathbf{S}_k, \mathbf{S}_{k+1}, X_k, Y_k) f(y_k|Y_k)\alpha_k(\mathbf{S}_k)\beta_k(\mathbf{S}_{k+1}) \\ & \sum_{X(\mathbf{S}_k,\mathbf{S}_{k+1})=X_k} h_k(\mathbf{S}_k, \mathbf{S}_{k+1}, X_k, Y_k(\mathbf{S}_k, \mathbf{S}_{k+1})) f(y_k|Y_k(\mathbf{S}_k, \mathbf{S}_{k+1}))\alpha_k(\mathbf{S}_k)\beta_k(\mathbf{S}_{k+1}) \end{aligned}$$

La probabilidad marginal a posteriori de X_k será el producto de los mensajes que recibe de los tres nodos-factores vecinos, con lo que obtenemos

$$\begin{aligned} P_{X_k}(X_k) &= f(x_k|X_k)p_k(X_k)m_k(X_k) \\ &= \sum_{X(\mathbf{S}_k,\mathbf{S}_{k+1})=X_k} \gamma_k(\mathbf{S}_k, \mathbf{S}_{k+1})\alpha_k(\mathbf{S}_k)\beta_k(\mathbf{S}_{k+1}) \end{aligned}$$

que corresponde a (2.4).

3.2. Cálculo en forma logarítmica. Desde el punto de vista numérico resulta más conveniente representar los mensajes mediante sus logaritmos $m(x) = \log \mu(x)$. Definimos

$$\begin{aligned} a_k(\mathbf{S}_k) &= \log \alpha_k(\mathbf{S}_k), \\ b_k(\mathbf{S}_k) &= \log \beta_k(\mathbf{S}_k), \\ g_k(\mathbf{S}_k, \mathbf{S}_{k+1}) &= \log \gamma_k(\mathbf{S}_k, \mathbf{S}_{k+1}). \end{aligned}$$

Definimos también la función

$$m^*(x_1, \ldots, x_n) = \log(e^{x_1} + \ldots + e^{x_n}).$$

De esta forma

- El mensaje de salida de cada nodo-variable pasa a ser la suma de los mensajes de entrada.
- El mensaje de salida de un nodo-factor de orden dos se calcula como

$$\begin{aligned} a_{k+1}(\mathbf{S}_{k+1}) &= \log \alpha_{k+1}(\mathbf{S}_{k+1}) \\ &= \log \sum_{\mathbf{S}_k} \gamma_k(\mathbf{S}_k, \mathbf{S}_{k+1}) \alpha_k(\mathbf{S}_k) \\ &= \log \sum_{\mathbf{S}_k} e^{g_k(\mathbf{S}_k, \mathbf{S}_{k+1})} e^{a_k(\mathbf{S}_k)} \\ &= \log \sum_{\mathbf{S}_k} e^{g_k(\mathbf{S}_k, \mathbf{S}_{k+1}) + a_k(\mathbf{S}_k)} \\ &= m^*_{\mathbf{S}_k} \left(g_k(\mathbf{S}_k, \mathbf{S}_{k+1}) + a_k(\mathbf{S}_k) \right). \end{aligned}$$

Para las variables binarias resulta más práctico representar los mensajes $\mu(x)$, $x = 0, 1$ mediante

$$L = \log \frac{\mu(0)}{\mu(1)} = m(0) - m(1).$$

Entonces los productos realizados en los nodos-variable corresponden a la suma de los valores recibidos:

$$\log \frac{\mu_1(0)\mu_2(0)}{\mu_1(1)\mu_2(1)} = \log \frac{\mu_1(0)}{\mu_1(1)} + \log \frac{\mu_2(0)}{\mu_2(1)}.$$

La operación inversa está dada por

$$L = \log \frac{p(0)}{p(1)} = \log \frac{p(0)}{1 - p(0)} \Rightarrow \mathbf{p} = [p(0), p(1)] = \frac{1}{e^{L/2} + e^{-L/2}} \left[e^{L/2}, e^{-L/2} \right]$$

y los mensajes generados por los nodos factor correspondientes a las observaciones son

$$L = \log \frac{f(x_i | X_i = 0)}{f(x_i | X_i = 1)} = 2 \frac{A}{\sigma^2} x_i$$

y análogamente para los y_i.

3.3. Código no sistemático no recursivo. Ahora consideramos el código no sistemático no recursivo dado por

$$\begin{aligned} U_j^0 &= X_j \oplus X_{j-1} \oplus X_{j-2}, \\ U_j^1 &= X_j \oplus X_{j-2}. \end{aligned} \tag{3.1}$$

o, con variables de estado,

$$\begin{aligned} U_k^0 &= X_k \oplus S_k^0 \oplus S_k^1, \\ U_k^1 &= X_k \oplus S_k^1, \\ S_{k+1}^0 &= X_k, \\ S_{k+1}^1 &= S_k^0, \end{aligned} \tag{3.2}$$

Notamos $\mathbf{S}_k = (S_k^0, S_k^1)$, $\mathbf{S} = (\mathbf{S}_k)$, $\mathbf{X} = (X_k)$, $\mathbf{U} = (U_k^0, U_k^1)$. Al ser $X_k = S_{k+1}^0$ prescindimos en adelante de las variables X_k. Como en el caso del otro codificador, la transición entre variables de estado determina los valores de los bits de salida,

$$\begin{aligned} U_k^0 &= U_k^0(\mathbf{S}_k, \mathbf{S}_{k+1}) = S_{k+1}^0 \oplus S_k^0 \oplus S_k^1, \\ U_k^1 &= U_k^1(\mathbf{S}_k, \mathbf{S}_{k+1}) = S_{k+1}^0 \oplus S_k^1, \end{aligned}$$

y podemos escribir la función de probabilidad conjunta como

$$P(\mathbf{S},\mathbf{U}) = \prod_{k=1}^{N} \eta_k(\mathbf{S}_k, \mathbf{S}_{k+1}, \mathbf{U}_k)$$

donde

$$\begin{aligned} \eta_k(\mathbf{S}_k, \mathbf{S}_{k+1}, \mathbf{U}_k) =& f\left(u_k^0 | U_k^0\right) f\left(u_k^1 | U_k^1\right) \\ & \underbrace{Z\left(U_k^0 \oplus S_{k+1}^0 \oplus S_k^0 \oplus S_k^1\right) Z\left(U_k^1 \oplus S_{k+1}^0 \oplus S_k^1\right)}_{W(\mathbf{S}_k, \mathbf{S}_{k+1}, \mathbf{U}_k)}. \end{aligned} \tag{3.3}$$

Notamos por $\alpha_k(\mathbf{S})$ los mensajes que envían los factores hacia abajo y por $\beta_k(\mathbf{S})$ los mensajes que envían hacia arriba. El mensaje del módulo superior al inferior es

$$\alpha_1(\mathbf{S}_1) = \gamma_1(\mathbf{0}, \mathbf{S}_1).$$

Este mensaje lo transfiere a la salida tal cual el nodo-variable $\mathbf{S}_1$. Un nodo genérico recibe el mensaje $\alpha_k(\mathbf{S}_k)$ y genera el mensaje

$$\begin{aligned} \alpha_{k+1}(\mathbf{S}_{k+1}) &= \sum_{\mathbf{S}_k} \eta_k\left(\mathbf{S}_k, \mathbf{S}_{k+1}, \mathbf{U}_k\left(\mathbf{S}_k, \mathbf{S}_{k+1}\right)\right) \alpha_k(\mathbf{S}_k) \\ &= \sum_{\mathbf{S}_k} f\left(u_k^0 | U_k^0\left(\mathbf{S}_k, \mathbf{S}_{k+1}\right)\right) f\left(u_k^1 | U_k^1\left(\mathbf{S}_k, \mathbf{S}_{k+1}\right)\right) \alpha_k(\mathbf{S}_k). \end{aligned}$$

Análogamente, el nodo-factor N genera el mensaje

$$\beta_N(\mathbf{S}_N) = \gamma_N(\mathbf{S}_N, \mathbf{0})$$

y un nodo-factor genérico k que reciba del nodo-variable el mensaje $\beta_{k+1}(\mathbf{S}_{k+1})$ generará hacia el nodo-variable $\mathbf{S}_k$ el mensaje

$$\begin{aligned}\beta_k(\mathbf{S}_k) &= \sum_{\mathbf{S}_k} \eta_k(\mathbf{S}_k, \mathbf{S}_{k+1}, \mathbf{U}_k)\beta_{k+1}(\mathbf{S}_{k+1}) \\ &= \sum_{\mathbf{S}_k} f\left(\mathbf{u}_k|\mathbf{U}_k\left(\mathbf{S}_k, \mathbf{S}_{k+1}\right)\right)\beta_{k+1}(\mathbf{S}_{k+1})\end{aligned}$$

donde

$$f\left(\mathbf{u}_k|\mathbf{U}_k\right) = f\left(u_k^0|U_k^0\right) f\left(u_k^1|U_k^1\right).$$

Este cálculo proporciona las probabilidades marginales a posteriori de los estados como

$$P_{\mathbf{S}_k}(\mathbf{S}_k) = \alpha_k(\mathbf{S}_k)\beta_k(\mathbf{S}_k).$$

Las probabilidades del bit de información $X_k = S_{k+1}^0$ se obtienen sumando respecto de S_{k+1}^1 en $P_{\mathbf{S}_{k+1}}(\mathbf{S}_{k+1})$.

En algunas aplicaciones (ver sección 6) es necesario generar mensajes desde cada factor $W(\mathbf{S}_k, \mathbf{S}_{k+1}, \mathbf{U}_k)$ hacia el nodo $\mathbf{U}_k$ correspondiente. Estos mensajes serán

$$\begin{aligned}r_k(\mathbf{U}_k) &= \sum_{\mathbf{S}_k, \mathbf{S}_{k+1}} \alpha_k(\mathbf{S}_k)\beta_{k+1}(\mathbf{S}_{k+1})W(\mathbf{S}_k, \mathbf{S}_{k+1}, \mathbf{U}_k) \\ &= \sum_{\mathbf{U}_k(\mathbf{S}_k, \mathbf{S}_{k+1})=\mathbf{U}_k} \alpha_k(\mathbf{S}_k)\beta_{k+1}(\mathbf{S}_{k+1}).\end{aligned} \tag{3.4}$$

3.4. Cálculo en forma logarítmica. Definimos

$$\begin{aligned}a_k(\mathbf{S}_k) &= \log \alpha_k(\mathbf{S}_k), \\ b_k(\mathbf{S}_k) &= \log \beta_k(\mathbf{S}_k), \\ F\left(u_k^0|U_k^0\right) &= \log f\left(u_k^0|U_k^0\right), \\ F\left(u_k^1|U_k^1\right) &= \log f\left(u_k^1|U_k^1\right), \\ F\left(\mathbf{u}_k|\mathbf{U}_k\right) &= \log f(\mathbf{u}_k|\mathbf{U}_k) = F\left(u_k^0|U_k^0\right) + F\left(u_k^1|U_k^1\right), \\ \rho_k\left(\mathbf{U}_k\right) &= \log r_k\left(\mathbf{U}_k\right).\end{aligned}$$

Tenemos

$$\begin{aligned}a_{k+1}(\mathbf{S}_{k+1}) &= \log \alpha_{k+1}(\mathbf{S}_{k+1}) \\ &= \log \sum_{\mathbf{S}_k} f\left(\mathbf{u}_k|\mathbf{U}_k\left(\mathbf{S}_k, \mathbf{S}_{k+1}\right)\right)\alpha_k(\mathbf{S}_k) \\ &= \log \sum_{\mathbf{S}_k} e^{F(\mathbf{u}_k|\mathbf{U}_k(\mathbf{S}_k, \mathbf{S}_{k+1}))} e^{a_k(\mathbf{S}_k)} \\ &= \log \sum_{\mathbf{S}_k} e^{F(\mathbf{u}_k|\mathbf{U}_k(\mathbf{S}_k, \mathbf{S}_{k+1}))+a_k(\mathbf{S}_k)} \\ &= m_{\mathbf{S}_k}^*\left(F\left(\mathbf{u}_k|\mathbf{U}_k\left(\mathbf{S}_k, \mathbf{S}_{k+1}\right)\right) + a_k(\mathbf{S}_k)\right),\end{aligned}$$

y, análogamente,

$$b_k(\mathbf{S}_k) = m_{\mathbf{S}_{k+1}}^*\left(F\left(\mathbf{u}_k|\mathbf{U}_k\left(\mathbf{S}_k, \mathbf{S}_{k+1}\right)\right) + b_{k+1}(\mathbf{S}_{k+1})\right).$$

Por otra parte, definiendo $\tilde{U} = (-1)^U$,

$$f(u_k^i|U_k^i) = \frac{1}{\sqrt{2\pi}\sigma} \exp - \frac{\left(u - A\tilde{U}_k^i\right)^2}{2\sigma^2}$$
$$\sim \exp\left(\frac{A}{\sigma^2} u_k^i \tilde{U}_k^i\right)$$

luego

$$F(u_k^i|U_k^i) = \log f_k(u_k^i|U_k^i) = \frac{A}{\sigma^2} u_k^i \tilde{U}_k^i.$$

El mensaje de cada factor W al nodo $\mathbf{U}_k$ correspondiente (3.4), en forma logarítmica, será

$$\rho_k(\mathbf{U}_k) = \log r_k(\mathbf{U}_k) = \log\left(\sum_{\mathbf{U}_k(\mathbf{S}_k,\mathbf{S}_{k+1})=\mathbf{U}_k} \alpha_k(\mathbf{S}_k)\beta_{k+1}(\mathbf{S}_{k+1})\right)$$
$$= m^*_{\mathbf{U}_k(\mathbf{S}_k,\mathbf{S}_{k+1})=\mathbf{U}_k}\left(a_k(\mathbf{S}_k) + b_{k+1}(\mathbf{S}_{k+1})\right). \tag{3.5}$$

Nota de implementación: La rutina que implementa los cálculos de esta subsección debe recibir como datos de entrada los valores $F(\mathbf{u}_k|\mathbf{U}_k)$. De esta forma será más fácil integrarla como parte del turboecualizador, donde mediante este parámetro se pasará el mensaje recibido del módulo de detección con IES.

4. Turbocódigos

La figura 4.1 representa el diagrama de un código turbo con concatenación en paralelo. La figura 4.2 incluye los factores correspondientes a las observaciones de los bits. El diagrama presenta bucles, por lo que no se puede aplicar el algoritmo suma-producto para obtener las probabilidades marginales a posteriori de los bits. Sin embargo la experiencia ha demostrado que aplicando el paso de mensajes en un cierto orden se obtiene un algoritmo iterativo que produce buenos resultados.

El algoritmo es el siguiente:

1. Consideramos el subgrafo a la izquierda de la línea de puntos 1 de la figura 4.2 *(corte 1)*. Aplicamos a este subgrafo el algoritmo suma-producto.
2. Consideramos el subgrafo a la derecha de la línea de puntos 2 *(corte 2)*. Aplicamos a este subgrafo el algoritmo suma-producto, pero incluyendo en él los mensajes generados en el apartado anterior propagados de izquierda a derecha sobre los arcos del corte 2.
3. Aplicamos de nuevo al subgrafo del apartado 1 el algoritmo suma producto, pero incluyendo en él los mensajes generados en el apartado anterior propagados de derecha a izquierda sobre los arcos del corte 1.
4. Iteramos cierto número de veces los apartados 2 y 3.
5. Para terminar calculamos las probabilidades marginales a posteriori de los bits de información multiplicando los mensajes recibidos por los nodos correspondientes por los tres arcos incidentes.

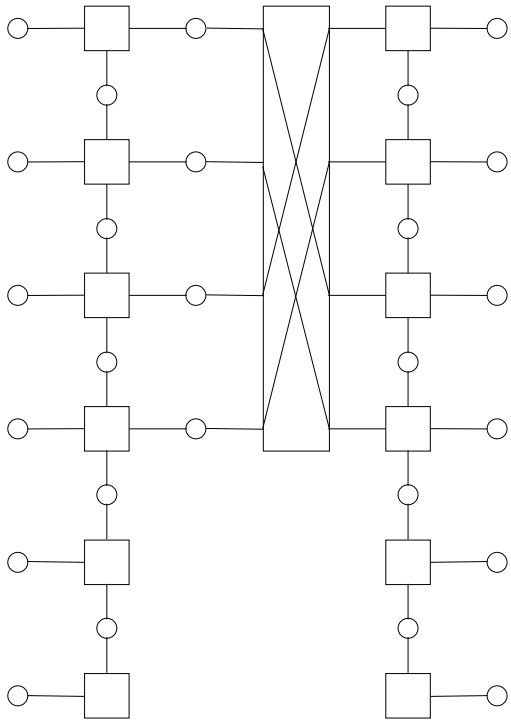

FIGURA 4.1. Diagrama de código turbo.

La figura 4.3 ilustra una interpretación de este algoritmo como iteración del algoritmo para la decodificación del algoritmo BCJR con probabilidades a priori de los bits de información. La operación descrita en el apartado 1 corresponde a la aplicación del algoritmo BCJR al grafo de la izquierda con equiprobabilidad a priori. El apartado 2 corresponde a la aplicación de este mismo algoritmo al grafo de la derecha con probabilidades a priori correspondientes a los mensajes generados en el apartado 1. Y el apartado 3 corresponde a la aplicación del algoritmo BCJR al grafo de la izquierda con probabilidades a priori correspondientes a los mensajes generados en el apartado 2.

5. Detección con IES

5.1. Esquema general. Consideramos la detección de una modulación 2-PAM sin codificar recibida por un canal con IES. donde f es la fdp de las muestras de ruido n_k,

$$f(n_k) = \frac{1}{\sqrt{2\pi}\sigma} \exp -\frac{n_k^2}{2\sigma^2}.$$

Suponemos el mismo el canal de [Berroux]. El diagrama de nuestro sistema está representado en la figura 6.1. Su formulación original es

$$z_k = \alpha_0 q\left(Y_k\right) + \alpha_1 \left(q\left(Y_{k-1}\right) + q\left(Y_{k+1}\right)\right) + \alpha_2 \left(q\left(Y_{k-2}\right) + q\left(Y_{k+2}\right)\right) + n_k. \tag{5.1}$$

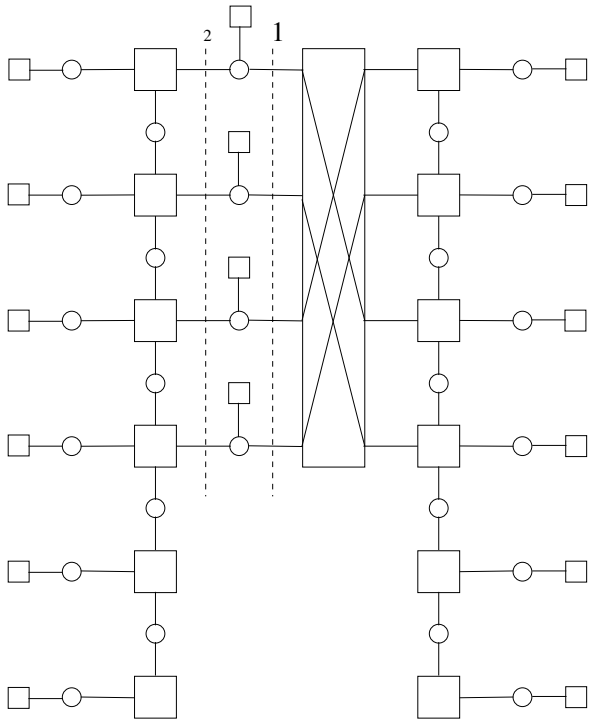

FIGURA 4.2. Diagrama de código turbo incluyendo observaciones de bits.

Nosotros desplazamos los índices de los Y_k para expresarlo de forma causal:

$$z_k = \alpha_0 q\left(Y_{k-2}\right) + \alpha_1\left(q\left(Y_{k-3}\right) + q\left(Y_{k-1}\right)\right) + \alpha_2\left(q\left(Y_{k-4}\right) + q\left(Y_k\right)\right) + n_k \tag{5.2}$$

y necesitamos variables de estado $\mathbf{T}_k = \left(T_k^0, T_k^1, T_k^2, T_k^3\right) = (Y_{k-1}, Y_{k-2}, Y_{k-3}, Y_{k-4})$, con las que el sistema queda de la forma

$$\begin{aligned}
z_k &= L\left(Y_k, \mathbf{T}_k\right) + n_k, \\
L\left(Y_k, \mathbf{T}_k\right) &= \alpha_0 q\left(T_k^1\right) + \alpha_1\left(q\left(T_k^2\right) + q\left(T_k^0\right)\right) \\
&\quad + \alpha_2\left(q\left(T_k^3\right) + q\left(Y_k\right)\right), \\
T_{k+1}^0 &= Y_k, \\
T_{k+1}^1 &= T_k^0, \\
T_{k+1}^2 &= T_k^1, \\
T_{k+1}^3 &= T_k^2
\end{aligned} \tag{5.3}$$

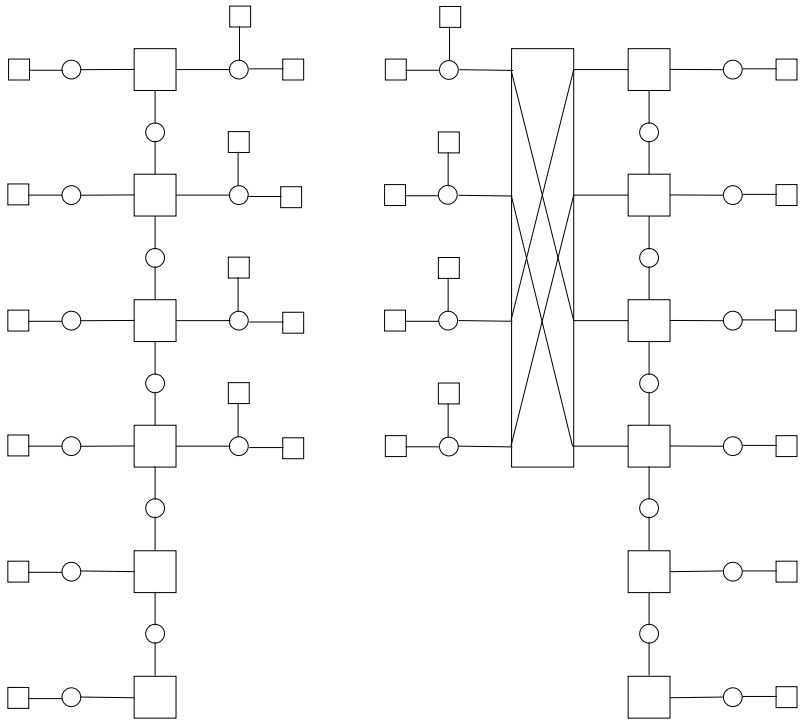

FIGURA 4.3. Algoritmo de decodificación de turbocódigos.

y los factores son de la forma

$$g_k(\mathbf{T}_k, \mathbf{T}_{k+1}, Y_k) = \underbrace{Z\left(T_{k+1}^0 \oplus Y_k^0\right) Z\left(T_{k+1}^1 \oplus T_k^0\right) Z\left(T_{k+1}^2 \oplus T_k^1\right) Z\left(T_{k+1}^3 \oplus T_k^2\right)}_{W(\mathbf{T}_k, \mathbf{T}_{k+1}, Y_k)} \cdot f\left(z_k - L\left(Y_k, \mathbf{T}_k\right)\right). \tag{5.4}$$

Tenemos además unos nodos factores hoja conectados a las variables Y_k, de la forma

$$p_k(Y_k).$$

Al tratarse de una red en forma de cadena el algoritmo es esencialmente el mismo que hemos visto para los decodificadores convolucionales. Además $\mathbf{T}_{k+1}$ determina Y_k, pues $Y_k = Y_k\left(\mathbf{T}_{k+1}\right) = T_{k+1}^0$.

El paso de mensajes de arriba a abajo, por ejemplo, quedaría

$$\begin{aligned}
\alpha_{k+1}(\mathbf{T}_{k+1}) &= \sum_{\mathbf{T}_k, Y_k} g_k\left(\mathbf{T}_k, \mathbf{T}_{k+1}, Y_k\right) p_k(Y_k) \alpha_k(\mathbf{T}_k) \\
&= \sum_{\mathbf{T}_k} g_k\left(\mathbf{T}_k, \mathbf{T}_{k+1}, Y_k\left(\mathbf{T}_{k+1}\right)\right) p_k(Y_k) \alpha_k(\mathbf{T}_k) \\
&= \sum_{\mathbf{T}_k} f\left(z_k - L\left(Y_k\left(\mathbf{T}_{k+1}\right), \mathbf{T}_k\right)\right) W\left(\mathbf{T}_k, \mathbf{T}_{k+1}, Y_k\left(\mathbf{T}_{k+1}\right)\right) p_k(Y_k) \alpha_k(\mathbf{T}_k).
\end{aligned}$$

Obsérvese que para cada $\mathbf{T}_{k+1}$ los valores de $\mathbf{T}_k$ para los que $W\left(\mathbf{T}_k, \mathbf{T}_{k+1}, Y_k\left(\mathbf{T}_{k+1}\right)\right)$ vale uno son sólo dos, puesto que $\mathbf{T}_{k+1}$ determina T_k^i, $i = 0, 1, 2$. Tenemos una ecuación similar para el paso de mensajes $\beta_k\left(\mathbf{T}_k\right)$ de abajo hacia arriba.

Además ahora hay que generar al final mensajes de los nodos factor g_k hacia las variables Y_k, de la forma

$$\mu(Y_k) = \sum_{\mathbf{T}_k, \mathbf{T}_{k+1}} \alpha_k\left(\mathbf{T}_k\right) \beta_{k+1}\left(\mathbf{T}_{k+1}\right) g_k(\mathbf{T}_k, \mathbf{T}_{k+1}, Y_k)$$

Esta fórmula parece que coincide con la de $L_e(x_{ij})$ en (11.22) de [Berroux], como debe ser, porque su γ_{i-1} incluye el término de probabilidades a priori $p_k(Y_k)$ en la expresión $\Pr(s|s')$ de (11.14).

5.2. Implementación logarítmica. Definimos

$$\begin{aligned} a_k(\mathbf{T}_k) &= \log \alpha_k(\mathbf{T}_k), \\ b_k(\mathbf{T}_k) &= \log \beta_k(\mathbf{T}_k), \\ F(L_k) &= \log f\left(z_k - L_k\right), \\ m(Y_k) &= \log \mu(Y_k) \end{aligned}$$

Tenemos

$$\begin{aligned} f\left(z_k - L_k\right) &= \frac{1}{\sqrt{2\pi}\sigma} \exp - \frac{\left(z_k - L_k\right)^2}{2\sigma^2} \\ &\sim \exp\left(-\frac{z_k^2 - 2L_k z_k + L_k^2}{2\sigma^2}\right) \\ &\sim \exp\left(-\frac{L_k^2 - 2L_k z_k}{2\sigma^2}\right) \end{aligned}$$

luego

$$F(L_k) = \log f\left(z_k - L_k\right) = \frac{L_k(2z_k - L_k)}{2\sigma^2}.$$

Tenemos

$$\begin{aligned} a_{k+1}(\mathbf{T}_{k+1}) &= \log \alpha_{k+1}(\mathbf{T}_{k+1}) \\ &= \log \sum_{\mathbf{T}_k} f\left(z_k - L\left(Y_k\left(\mathbf{T}_{k+1}\right), \mathbf{T}_k\right)\right) W\left(\mathbf{T}_k, \mathbf{T}_{k+1}, Y_k\left(\mathbf{T}_{k+1}\right)\right) p_k(Y_k) \alpha_k(\mathbf{T}_k) \\ &= m^*_{\mathbf{T}_k(\mathbf{T}_{k+1})}\left(F\left(L\left(Y_k\left(\mathbf{T}_{k+1}\right), \mathbf{T}_k\right)\right) + \log p_k(Y_k) + a_k\left(\mathbf{T}_k\right)\right) \end{aligned}$$

donde $\mathbf{T}_k(\mathbf{T}_{k+1})$ representa el conjunto de valores de $\mathbf{T}_k$ compatibles con el valor de $\mathbf{T}_{k+1}$ (dos valores como hemos indicado anteriormente).

Para la generación de los mensajes $b_k(\mathbf{T_k})$ el procedimiento es análogo.

En cuanto a los mensajes hacia los nodos Y_k,

$$
\begin{aligned}
m(Y_k) &= \log \mu(Y_k) \\
&= \log \left(\sum_{\mathbf{T}_k, \mathbf{T}_{k+1}} \alpha_k \left(\mathbf{T}_k \right) \beta_{k+1} \left(\mathbf{T}_{k+1} \right) g_k(\mathbf{T}_k, \mathbf{T}_{k+1}, Y_k) \right) \\
&= \log \left(\sum_{\mathbf{T}_k, \mathbf{T}_{k+1}} \alpha_k \left(\mathbf{T}_k \right) \beta_{k+1} \left(\mathbf{T}_{k+1} \right) f \left(z_k - L \left(Y_k \left(\mathbf{T}_{k+1} \right), \mathbf{T}_k \right) \right) W \left(\mathbf{T}_k, \mathbf{T}_{k+1}, Y_k \left(\mathbf{T}_{k+1} \right) \right) \right) \\
&= m^*_{(\mathbf{T}_k, \mathbf{T}_{k+1})(Y_k)} \left(a_k \left(\mathbf{T}_k \right) + b_{k+1} \left(\mathbf{T}_{k+1} \right) + F \left(L \left(Y_k \left(\mathbf{T}_{k+1} \right), \mathbf{T}_k \right) \right) \right)
\end{aligned}
$$

donde $(\mathbf{T}_k, \mathbf{T}_{k+1})(Y_k)$ representa el conjunto de los ocho posibles valores distintos del par $(\mathbf{T}_k, \mathbf{T}_{k+1})$ compatible con un cierto valor de Y_k.

Nota de implementación: La rutina que implementa los cálculos de esta sección debe recibir como datos de entrada los valores $\log p_k(Y_k)$. De esta forma será más fácil integrarla como parte del turboecualizador, donde mediante este parámetro se pasará el mensaje recibido del módulo de decodificación del turbocódigo.

6. TURBOECUALIZACIÓN

6.1. Esquema general. Consideramos un sistema de transmisión constituido por el encadenamiento de un codificador convolucional (CC) NS-NR, operando por bloques, una permutación y un sistema 2-PAM que transmite bits equiprobables. El objetivo es realizar la decodificación óptima aproximada en el receptor mediante un modelo de red bayesiana.

Notamos por X_j los bits de información, por U_j^l los bits de salida del CC, por Y_k sus versiones multiplexadas y permutadas y por Z_k las amplitudes recibidas. Tenemos para el codificador la relación (3.1) y para el conjunto modulador-canal-demodulador la indicada en la ecuación (5.2). Introduciendo variables de estado tenemos para el codificador las ecuaciones (3.2) y para el segundo las indicadas en (5.3).

La función de probabilidad es el producto de factores correspondientes a cada uno de estos grupos de ecuaciones más la correspondencia entre los U_j^i y los Y_k dada por la permutación.Los factores asociados al código convolucional son los indicados en (3.3).Los factores correspondiente a las ecuaciones de la observación están especificados en la ecuación (5.4). Las figuras 6.1 y 6.2 representan los grafos factoriales correspondientes.

La figura 6.3 muestra la red global y su descomposición en dos subredes sin bucles para la aplicación del algoritmo suma-producto. El algoritmo es equivalente a iterar las operaciones sobre grafos en forma de cadena

- Detección con IES (sección 5)
- Decodificación convolucional (sección 3.4)

En este esquema equivalente la información que pasa del primer módulo al segundo se transmite en forma de observaciones en el segundo , y la información del segundo al primero en forma de probabilidades a priori de los bits.

Recordamos que la secuencia formada por las variables Y_k y la formada por las variables $\mathbf{U}_k = \left(U_k^0, U_k^1 \right)$ están relacionadas por una permutación.

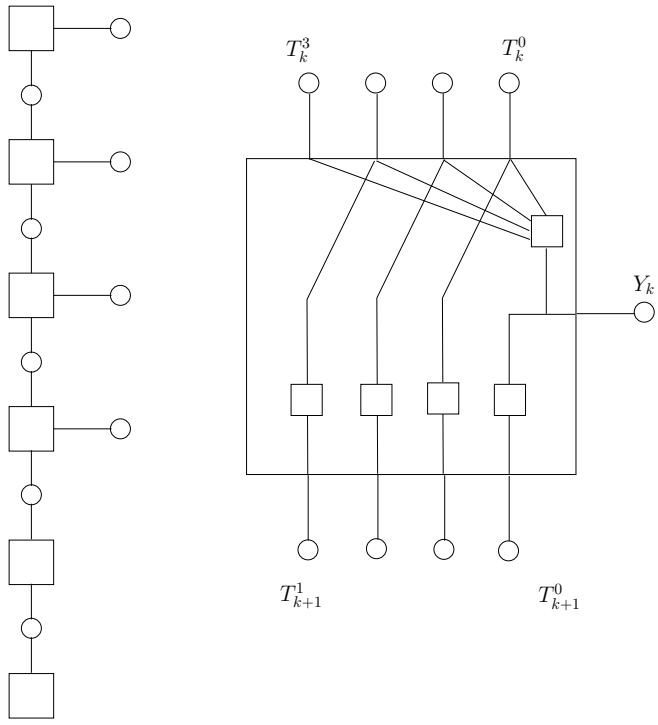

FIGURA 6.1. Grafo factorial de la detección con IES.

6.2. Evaluación de prestaciones. Para evaluar la probabilidad de error en función de la energía por bit normalizada hay que tener en cuenta que la señal recibida es

$$q[n] = x[n] * h[n]$$

donde $x[n]$ es una señal de muestras independientes de valor ± 1. Por tanto, $R_x[m] = \delta[n]$,

$$R_q[m] = R_x[m] * h[m] * h[-m] = h[m] * h[-m],$$

y

$$E\left[q[n]^2\right] = R_q[0] = \sum_m h[m]^2.$$

REFERENCIAS

[Berroux] C. Berroux (Ed.), Codes and turbo codes. Springer, 2010.
[Bishop] C. M. Bishop, Pattern recoginition and machine learning. Cambridge Univ. Press, 2006
[Forney] Principles of Digital Communications II, MIT Opencourseware.
[Madhow] U. Madhow, Fundamentals of Digital Communications. Cambridge Univ. Press, 2008.
[Wiberg] N. Wiberg, Codes and decoding on general graphs, Ph. D. thesis, Linköping Univ., 1995.

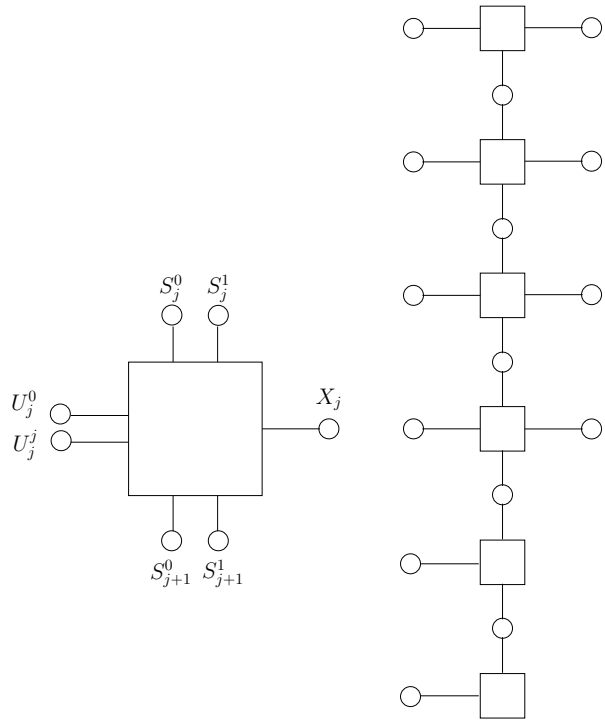

FIGURA 6.2. Grafo factorial del codificador convolucional utilizado en el esquema de turboecualización.

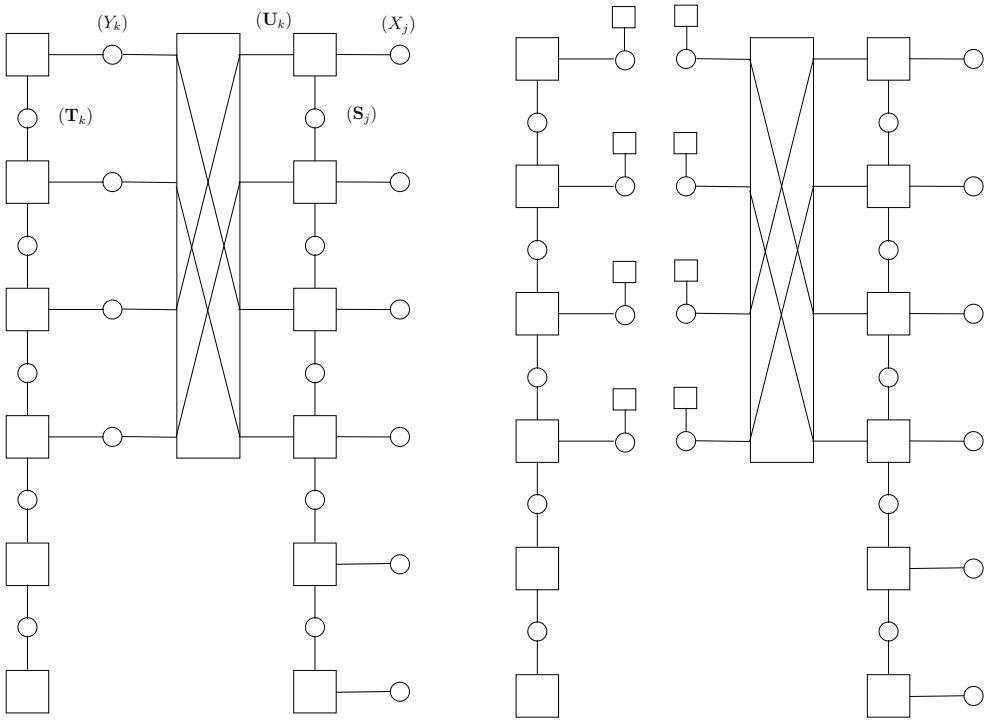

FIGURA 6.3. Grafo factorial del turboecualizador y su descomposición en dos subgrafos.